CLAIMS

What is claimed is:

1. A TCP offload engine (TOE) capable of offloading, from a host, TCP protocol processing tasks that are associated with a TCP connection, the TOE comprising:

a first memory that stores and simultaneously outputs at least two TCP state values associated with the TCP connection, wherein the at least two TCP state values are taken from the group consisting of: a receive packet sequence limit, an expected receive packet sequence number, a transmit sequence limit, a transmit acknowledge number, and a transmit sequence number;

a second memory that stores and simultaneously outputs at least two packet header values of a packet communicated over the TCP connection, wherein the at least two packet header values are taken from the group consisting of: a receive packet sequence number, a packet payload size, a packet acknowledge number, and a packet transmit window value; and

combinatorial logic that receives said at least two TCP state values and said at least two packet header values and generates therefrom a flush detect signal indicative of whether an error condition has occurred, the two TCP state values and the two packet header values all being supplied to the combinatorial logic simultaneously.

2. The TOE of Claim 1, wherein the TOE transfers the receive packet sequence limit, the expected receive packet sequence number, the transmit sequence limit, the transmit acknowledge number, and the transmit sequence number to the host if the flush detect signal is indicative of the error condition.

3. The TOE of Claim 1, wherein more than two values of the group of TCP state values are supplied simultaneously to the combinatorial logic, and wherein more than two values of the group of TCP state values are used to generate the flush detect signal.

4. The TOE of Claim 1, wherein more than two values of the group of packet header values are supplied simultaneously to the combinatorial logic, and wherein more than two values of the group of packet header values are used to generate the flush detect signal.

5. The TOE of Claim 1, wherein the TOE can control a plurality of TCP connections, each of said TCP connections being associated with a TCB identifier, and wherein the first memory comprises a plurality of transaction control blocks (TCB), wherein the first memory can be addressed by a TCB identifier to access a TCB that contains TCP state information for a TCP connection associated with the TCB identifier.

6. The TOE of Claim 1, wherein the combinatorial logic is part of a state machine, the state machine being clocked by a clock signal, wherein the combinatorial logic generates said flush detect signal from said at least two TCP state values and said at least two packet header values within approximately one period of the clock signal.

7. The TOE of Claim 6, wherein the state machine does not fetch instructions, decode the instructions, and execute the instructions.

8. The TOE of Claim 7, wherien the state machine causes the expected packet receive sequence number and the receive packet sequence limit values in the first memory to be updated in a single period of the clock signal.

9. The TOE of Claim 8, wherein the TCP connection was set up by a stack executing on the host, and wherein control of the TCP connection was then passed from the host to the TOE.

10. The TOE of Claim 7, wherein a packet having a payload is received onto the TOE, the TOE further comprising:

a DMA controller that moves the payload from the TOE to the host using a source address value, a size value indicating an amount of information to move, and a destination address value, wherein the state machine causes the source address value, the

size value, and the destination address value to be supplied to the DMA controller in a single state of the state machine.

11. A TCP offload engine (TOE) capable of offloading TCP protocol processing tasks from a host, the TCP protocol processing tasks being associated with a TCP connection, the TOE comprising:

a first memory that stores and simultaneously outputs at least two TCP state values associated with the TCP connection, wherein the at least two TCP state values are taken from the group consisting of: a receive packet sequence limit, an expected receive packet sequence number, a transmit sequence limit, a transmit acknowledge number, and a transmit sequence number;

a second memory that stores and simultaneously outputs at least two packet header values of a packet communicated over the TCP connection, wherein the at least two packet header values are taken from the group consisting of: a receive packet sequence number, a packet payload size, a packet acknowledge number, and a packet transmit window value; and

a hardware state machine that receives said at least two TCP state values and said at least two packet header values and generates therefrom a signal indicative of whether an exception condition has occurred, the two TCP state values and the two packet header values all being supplied to the hardware state machine simultaneously, wherein the hardware state machine is clocked by a clock signal, wherein the hardware state machine generates said signal from said at least two TCP state values and said at least two packet header values within approximately one period of the clock signal.

12. The TOE of Claim 11, wherein the hardware state machine causes the expected packet receive sequence number and the receive packet sequence limit values in the first memory to be updated in a single period of the clock signal.

13. The TOE of Claim 12, wherein the expected packet receive sequence number and the receive packet sequence limit values are updated by simultaneously writing the expected packet receive sequence number value and the receive packet sequence limit value into the first memory.

14. The TOE of Claim 11, wherein the exception condition is a condition which results in control of the TCP connection being passed from the TOE to the host.

15. The TOE of Claim 11, wherein the first memory is a dual-port memory that has a first interface and a second interface, the hardware state machine reading from and writing to the first memory via the first interface, and wherein information passes from the host and into the first memory via the second interface.

16. The TOE of Claim 15, wherein the first memory comprises a plurality of TCB buffers, wherein one of the TCB buffers is associated with the TCP connection, and wherein a memory descriptor list entry associated with said TCP connection is stored in said TCB buffer associated with said TCP connection.

17. The TOE of Claim 11, further comprising:

a third memory that stores a plurality of socket descriptors, wherein one of the socket descriptors is associated with the TCP connection.

18. The TOE of Claim 11, wherein the second memory outputs a parse value along with said at least two packet header values, said parse value being indicative of whether the transport and network layer protocols of an associated packet are the TCP and IP protocols.

19. A TCP offload engine (TOE) capable of offloading TCP protocol processing tasks from a host, the TCP protocol processing tasks being associated with a TCP connection, the TOE comprising:

a memory that simultaneously outputs at least two TCP state values associated with the TCP connection, wherein said at least two TCP state values are taken from the group consisting of: a receive packet sequence limit, an expected receive packet sequence number, a transmit sequence limit, a transmit acknowledge number, and a transmit sequence number, the memory also simultaneously outputting at least two packet header values of a packet communicated over the TCP connection, wherein said at least two

packet header values are taken from the group consisting of: a receive packet sequence number, a packet payload size, a packet acknowledge number, and a packet transmit window value; and

means for receiving said at least two TCP state values and said at least two packet header values and for generating therefrom a signal indicative of whether an exception condition has occurred, the two TCP state values and the two packet header values all being supplied by the memory and to the means simultaneously.

20. The TOE of Claim 19, wherein the memory comprises a first memory portion and a second memory portion, the first memory portion storing said at least two TCP state values, the second memory portion storing said at least two packet header values.

21. The TOE of Claim 19, wherein the means is also for updating TCP state values by writing TCP state values into the memory.

22. The TOE of Claim 19, wherein the means comprises no sequential logic elements, the means consisting entirely of combinatorial logic elements.

23. The TOE of Claim 22, wherein the means is a part of a state machine.

24. The TOE of Claim 19, wherein the exception condition is a condition that results in control of the TCP connection being passed from the TOE to the host.